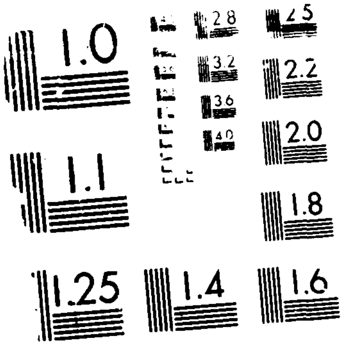
AD-A188 486   RESEARCH AND DEVELOPMENT IN NATURAL LANGUAGE   1/1
UNDERSTANDING AS PART OF THE. (U) BBN LABS INC
CAMBRIDGE MA R M WEISCHEDEL ET AL APR 87 BBN-6463
UNCLASSIFIED N00014-85-C-0016   F/G 12/5   NL


1.0
2.8
2.5
3.2
2.2
3.6
4.0
2.0
1.1
1.8
1.25
1.4
1.6

(2)

BBN Laboratories Incorporated

A Subsidiary of Bolt Beranek and Newman Inc.

Report No. 6463

Research and Development in Natural Language Understanding as Part of the Strategic Computing Program

Annual Technical Report
December 1984 — December 1985

R. Weischedel, R. Scha, E. Walker, D. Ayuso, A. Haas, E. Hinrichs, R. Ingria, L. Ramshaw, V. Shaked, D. Stallard, J. de Bruin, and K. Koile

Submitted to:
DARPA

REPORT DOCUMENTATION PAGE

READ INSTRUCTIONS
BEFORE COMPLETING FORM

1. REPORT NUMBER
6463

2. GOVT ACCESSION NO.

3. RECIPIENT'S CATALOG NUMBER

4. TITLE (and Subtitle)
Research and Development in Natural Language Understanding as Part of the Strategic Computing Program
Annual Technical Report 12/84-12/85

5. TYPE OF REPORT & PERIOD COVERED
Annual Report
12/84-12/85

6. PERFORMING ORG. REPORT NUMBER
6463

7. AUTHOR(s) R. Weischedel, R. Scha, E. Walker, D. Ayuso, A. Haas, E. Hinrichs, R. Ingria, L. Ramshaw, V. Shaked, D. Stallard, J. de Bruin, and K. Koile

8. CONTRACT OR GRANT NUMBER(s)
N00014-85-0016

9. PERFORMING ORGANIZATION NAME AND ADDRESS
BBN Laboratories Inc.
10 Moulton Street
Cambridge, MA 02238

10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS

11. CONTROLLING OFFICE NAME AND ADDRESS
Office of Naval Research
Department of the Navy
Arlington, VA 22217

12. REPORT DATE
April 1987

13. NUMBER OF PAGES
40

14. MONITORING AGENCY NAME & ADDRESS(if different from Controlling Office)

15. SECURITY CLASS. (of this report)
Unclassified

15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)
IRUS, Natural Language Interfaces, Knowledge Representation, NIKL, Knowledge Acquisition, Semantics, Battle Management

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

This report summarizes BBN's first year's activity in research and development in natural language processing within DARPA's Strategic Computing Program.

The first chapter summarizes the direction in this work, including not only the areas of our own research and development but also possible integration of components and techniques from other sites. The second chapter reviews our activities in technology transfer, namely, moving a natural language interface developed (cont.)

20. (Continued)

over a period of years of basic research out of the laboratory. The software is being used by The Naval Ocean Systems Center in conjunction with BBN to provide a prototype natural language interface at The Pacific Fleet Command Center. The third chapter describes a simple inference procedure which has proved quite effective in reducing the semantic representation of the user's input to terms of the underlying system(s).

Table of Contents

RESEARCH AND DEVELOPMENT IN NATURAL LANGUAGE UNDERSTANDING AS PART OF THE STRATEGIC COMPUTING PROGRAM

Annual Report
21 December 1984 - 20 December 1985

Principal Investigator:
Dr. Ralph M. Weischedel

Prepared for:

Defense Advanced Research Projects Agency
1400 Wilson Boulevard
Arlington, VA 22209

ARPA Order No. 5257	Contract No. N00014-85-C-0016
Effective Date of Contract 21 December 1984	Contract Expiration Date 28 December 1987
Amount of Contract: $2,648,558.00	Scientific Officer Dr. Alan R. Meyrowitz

This research was supported by the Advanced Research Projects Agency of the Department of Defense and was monitored by ONR under Contract No. N00014-85-C-0016. The views and conclusions contained in this document are those of the authors and should not be interpreted as necessarily representing the official policies, either expressed or implied, or the Defense Advanced Research Projects Agency or the U.S. Government.

1. Research and Development in Natural Language Processing at BBN Laboratories in the Strategic Computing Program[1]

Ralph Weischedel, Remko Scha, Edward Walker,
Damaris Ayuso, Andrew Haas, Erhard Hinrichs, Robert Ingria,
Lance Ramshaw, Varda Shaked, David Stallard

1.1 Background

BBN's responsibility is to conduct research and development in natural language interface technology. This responsibility has three aspects:

- to demonstrate state-of-the-art technology in a Strategic Computing application, collecting data regarding the effectiveness of the demonstrated heuristics,
- to conduct research in natural language interface technology, as itemized in the description of JANUS *later in this note, and*
- to integrate technology from other natural language interface contractors, including USC/Information Sciences Institute, the University of Pennsylvania, and the University of Massachusetts.

The Fleet Command Center Battle Management Program (FCCBMP) has been the application providing the domain in which our work is being carried out. The FCCBMP encompasses the development of expert system capabilities at the Pacific Fleet Command Center in Hawaii, and the development of an integrated natural language interface to these new capabilities as well as to the existing data bases and graphic display facilities. BBN is developing a series of increasingly sophisticated natural language understanding systems which will serve as an integrated interface to several facilities at the Pacific Fleet Command Center: the Integrated Data Base (IDB), which contains information about ships, their readiness states, their capabilities, etc.; the Operations Support Group Prototype (OSGP), a graphics system which can display locations and itineraries of ships on maps; and the Force Requirements Expert System (FRESH) which is being built by Texas Instruments.

The target users for this application are naval officers involved in decision making at the Pacific Fleet Command Center; these are executives whose effort is better spent on navy problems and decision making than on the details of which software system offers a given information capability, how a problem should be divided to make use of the various systems, or how to synthesize the results from several sources into the desired answer. Currently they do not access the data base or OSGP application programs themselves; instead, on a round-the-clock basis, two

[1] This paper was originally published in "The FINITE STRING Newsletter" in *Computational Linguistics*, Volume 12, April-June 1986. Requests for copies should be addressed to: Dr.Donald E. Walker (ACL), Bell Communications, Research, 435 South Street MRE 2A379, Morristown, NJ 07960, USA

operators act as intermediaries between the Navy staff and the computers. The utility of a natural language interface in such an environment is clear.

The starting point for development of the natural language interface system at the Pacific Fleet Command Center was the IRUS system, which has been under development at BBN for a number of years. A new version of this system, IRUS-86, has been installed in the FCCBMP testbed area at the Pacific Fleet Command Center for demonstration. Further basic research on the problems of natural language interfacing is continuing, and the results of this and future research will be incorporated into a next generation natural language interface system called JANUS, to be delivered to the Pacific Fleet Command Center at a later date. JANUS will share most of its domain-dependent data with IRUS-86, and it will share other modules as well; IRUS-86 will therefore be able to evolve gradually into the final version of JANUS.

1.2 IRUS-86: The Initial Test Bed System

The architecture of IRUS [2] is a cascade consisting of a sequence of translation modules:

- An ATN parser which produces a syntactic tree.
- A semantic interpreter which produces a formula of the meaning representation language MRL.
- A postprocessor for resolving anaphora and ellipsis.
- A translation module which produces a formula of the relational data base language ERL ("Extended Relational Language").
- A translation module which produces a sequence of commands for the underlying data base access system.

IRUS-86, the version of IRUS which is now installed at the Pacific Fleet Command Center, is a version of IRUS which is extended in several ways. Two of these extensions are especially worth mentioning:

- IRUS-86 uses the NIKL system [8] to represent its domain model, i.e., the relationships between the predicates and relations of the meaning representation language MRL. The NIKL domain model supports the system's treatment of semantic anomaly, anaphora, and nominal compounds.
- IRUS-86 contains a new module which exploits this NIKL domain model to simplify MRL expressions; this makes it possible to translate complex MRL-expressions into ERL constants, thus allowing for significant divergences between the input English and the structure of the underlying data base [13].

In addition to accessing the NIKL domain model, the parser, semantic interpreter and MRL-to-ERL translator access other knowledge sources which contain domain-dependent information:

- the lexicon,
- the semantic interpretation rules for individual concepts,
- the MRL-to-ERL mapping rules for individual MRL constants, which introduce the details of underlying system structure, such as file and field names.

To port IRUS to the navy domain, the relevant domain-dependent data had to be supplied to the system. This task is being accomplished by personnel at the Naval Ocean Systems Center (NOSC). In August, 1985, BBN provided NOSC with an initial prototype system containing small example sets of lexical entries, semantic interpretation rules, and MRL-to-ERL rules; using acquisition tools provided by BBN, NOSC personnel have been entering the rest of the data.

IRUS-86 was delivered to the FRESH developers at Texas Instruments in January 1986, was installed in a test bed area of the Pacific Fleet Command Center in April 1986, and will be demonstrated in June 1986. Currently, the lexicon and the domain-dependent rules of the system only cover a relatively small part of the OSGP capabilities and the files and attributes of the Integrated Data Base. Once enough data have been entered so that the system covers a sufficiently large part of the data base, it will be tried out in actual use by Navy personnel. This will enable us to gather data about the way the system performs in a real environment, and to fine-tune the system in various respects. For instance, IRUS-86 makes use of shallow heuristic methods to address some aspects of natural language understanding such as anaphora and ellipsis for which general solutions are still research issues. The FCCBMP application provides a test bed in which such heuristic methods can be evaluated, and enhancements to them developed and tested, as part of the evolutionary technological growth intended to continue throughout the Natural Language Technology effort of the Strategic Computing Program.

1.3 Functional Goals for JANUS

The IRUS-86 system excels by its clean, modular structure, its broad syntactic/semantic coverage, its sophisticated domain model, and its systematic treatment of discrepancies between the English lexicon and the data base structure. We thus expect that it will demonstrate considerable utility as an interface component in the FCCBMP application. Nevertheless, IRUS-86 shares with other current systems several limitations which should be overcome if natural language interfaces are to become truly "natural". In developing JANUS, the successor of IRUS-86, we shall attempt to overcome some of those limitations. The areas of increased functionality we are considering are: semantics and knowledge representation, ill-formedness, discourse, cooperativeness, multiple underlying systems, and knowledge acquisition.

1.3.1 Semantics and Knowledge Representation

IRUS-86, like most other current systems, represents sentence meanings as formulas of a logical language which is a slight extension of first-order logic. As a consequence, many important phenomena in English have no equivalent in the meaning representation language, and cannot be dealt with correctly, e.g., modalities, propositional attitudes, generics, collective quantification, and context-dependence. Thus, one foregoes one of the most important potential assets of a natural language interface: the capacity of expressing complex semantic structures in a succinct and comfortable way.

In JANUS, we will therefore adopt a new meaning representation language which combines features from PHLIQA1's enriched lambda-calculus [10] with ideas underlying Montague's Intensional Logic [7], and possibly a distributed quote-operator [5]. It will have sufficient expressive power to incorporate a version of Carlson's treatment of generics [3], a version of Scha's treatment of quantification [11], Montague's treatment of modality, and various possible approaches to propositional attitudes and context-dependence.

In adopting a higher order logic as proposed, one confronts problems of formula simplification and the need to apply meaning postulates to reduce the semantic representation of an input sentence to an expression appropriate to the underlying system, e.g., a relational algebra expression in the case that the underlying system is a data base. To do this, we will investigate the limited inference mechanisms of KL-TWO [8, 14], following up on our previous work [13]. The advantage of these inference mechanisms is their tractability; discovering their power and limitations in this complex problem domain should be an interesting result.

1.3.2 Discourse

The meaning of a sentence depends in many ways on the context which has been set up by the preceding discourse. IRUS and other systems, however, currently ignore most of these dependencies, and employ a rather shallow model of discourse structure. To allow the user to exploit the full expressive potential of a natural language interaction, the system must track topics, reference times, possible antecedents for anaphora, etc.; it must be able to recognize the constituent units of a discourse and the subordination or coordination relations obtaining between them. A substantial amount of work has been done already on several of these issues, much of it by BBN researchers [12, 6, 9, 4]. Research in this area continues under a separate DARPA-funded contract. We expect to be able to integrate some of the results of that research in the JANUS system.

1.3.3 Ill-formedness

A natural interface system should be forgiving of a user's deviations from its expectations, be they misspellings, typographical errors, unknown words, poor syntax, incorrect presuppositions, fragmentary forms, or violated selection restrictions. Empirical studies show that as much as 25% of the input to data base query systems is ill-formed.

IRUS currently handles some classes of ill-formedness by using a combination of shallow heuristics and user interaction. It can correct for typographical misspellings, for omitted determiners or prepositions, and for some ungrammaticalities, like determiner-noun and subject-verb disagreement. The JANUS system will employ a more general approach to ill-formedness that will handle a larger class of ungrammatical constructions and a larger class of word selection problems, and that will also explore correcting several types of semantic ill-formedness.

These capabilities have major implications for the control of the understanding process, since considering such possibilities can exponentially expand the search space. Maintaining control will require care in integrating the ill-formedness capability into the rest of the system, and also making maximal use of the guidance that can be derived from a model of the discourse and user's goals to constrain the search.

1.3.4 Cooperativeness

A truly helpful system should not react to the literal meaning of a sentence, but to its perceived intent. If in the context of a given application it is possible to characterize the goals that a user may be expected to be pursuing through his interaction with the system, the system should try to infer from the user-input what the underlying goal could be. A system can do this by accessing a goal-subgoal hierarchy which links the speech acts expressed by individual utterances to the global goals that the user may have. This strategy has been applied successfully to rather small domains [1, 12]. We wish to investigate whether it carries over to the FCCBMP applications.

1.3.5 Modelling the Capabilities of Multiple Systems

The way in which IRUS-86 decides whether an input sentence translates into an IDB query or an OSGP command may be refined. There is a need for work on what kind of knowledge would be necessary to interface smoothly and intelligently to multiple underlying systems. A reasoning component is needed that can determine which underlying system or systems can best fulfill a user's request. Such a reasoning component would have to combine a model of the capabilities of the underlying systems with a model of the user goals and current intentions in the discourse context in order to choose the correct system(s). Such a model would also be useful for providing supporting information to the user.

1.3.6 Knowledge Acquisition

Further research is also called for to expand the power of the knowledge acquisition tools that are used in adding to the lexicon, the set of case frame rules, the model of domain predicates, and the set of transformation rules between the Meaning Representation Language and the languages of the underlying systems. The acquisition tools available in IRUS, unlike those in some other systems, are not tied to the specific fields and relations in the underlying database. The acquisition tools should work on the higher level of the domain model, since that provides a more general and transportable result. The knowledge acquisition facilities for JANUS will also need to be redesigned to support and to make maximal use of the power of the new meaning representation language based on intensional logic.

1.4 New Underlying Technologies

1.4.1 Coping with Ambiguity

The new functionalities we described in the previous section, and the techniques we intend to use to achieve them, raise an issue which has important consequences for the design of JANUS: we will be faced with an explosion in the number of interpretations that the system will have to process; every sentence will be manifold ambiguous. One source of this phenomenon is the improvement of the semantic coverage and the broadening of the discourse context. Distinctions and ambiguities which so far were ignored will be dealt with: for instance, different interpretation and scopes of quantifiers will be considered, and different antecedents for pronouns. Even more serious is the processing of ill-formed sentences, which may require trying out all partial interpretations to see which one can be extended to a complete interpretation after relaxing one or more constraints.

To cut down on the processing of spurious interpretations, it is very important that interpretations of sentences and their constituents be tested for plausibility at an early stage. Different techniques must probably be used in conjunction:

- Simplification transformations may show that an interpretation is absurd, by reducing it to TRUE or FALSE or the empty set.
- The discourse context and the model of the user's goals impose constraints on expected sentences.

1.4.2 Parallel Parsing

Since some of the techniques that we intend to use to fight the ambiguity explosion are themselves rather computation-intensive, it is clearly unavoidable that the improved system functionality that we aim for will lead to a considerable increase in the amount of processing required. To avoid a serious decrease of the new system's response times, we will therefore move it to a suitable parallel machine such as BBN's Butterfly or Monarch, running a parallel Common Lisp. This in itself has rather serious consequences for the software design. It means that from the outset we will keep parallelizability of the software in mind.

We have begun to address this issue in the area of syntax. A new declarative grammar is being written, which will ultimately have a coverage of English larger than the current RUS grammar; the grammar is written in a side-effect-free formalism (a context-free grammar with variables) so that different parsing algorithms may be explored which are easily parallelizable. The first such algorithm was implemented in May 1986 on <BBN's Butterfly.

1.5 Contributions from Other Sites

1.5.1 ISI UMass: Generation

We should not expect that JANUS will always be able to assess correctly which interpretation of a sentence is the intended one. In light of such situations, it is very important that the system can give a paraphrase of the input to the user, which shows the system's interpretation. This may be done either explicitly or as part of the answer. To be able to develop such capabilities, work on Natural Language Generation is needed. At USC/ISI a project directed by William Mann and Norman Sondheimer is underway to develop the generation system PENMAN, using the NIGEL systemic grammar. PENMAN will be integrated to become the generation component of JANUS. PENMAN itself consists of several subcomponents. Some of these, specifically the "text planning" component, will be developed through joint work between USC/ISI and David McDonald at the University of Massachusetts, based on the latter's experience with the MUMBLE system.

1.5.2 UPenn: Cooperation and Clarification

Under the direction of Aravind Joshi and Bonnie Webber at the University of Pennsylvania, several focused studies have been carried out to investigate various aspects of cooperative system behavior and clarification interactions. As part of the Strategic Computing Natural Language effort, UPenn will eventually develop this into a module which can be integrated into JANUS to further enhance its capabilities.

References

[1] Allen. J.F. Recognizing Intentions from Natural Language Utterances. In M. Brady and R.C. Berwick (editors), *Computational Models of Discourse*, pages 107-166. MIT Press, Cambridge, MA. 1983.

[2] Bates. M. and Bobrow, R.J. A Transportable Natural Language Interface for Information Retrieval. In *Proceedings of the 6th Annual International ACM SIGIR Conference*. ACM Special Interest Group on Information Retrieval and American Society for Information Science, Washington, D.C., June, 1983.

[3] Carlson, G. *Reference to Kinds in English*. Garland Press, New York, 1979.

[4] Grosz, B.J. and Sidner, C.L. The Structures of Discourse Structure. In L. Polanyi (editor), *Discourse Structure*. Ablex Publishers, Norwood, NJ, 1986.

[5] Haas, A.R. A Syntactic Theory of Belief and Action. *Artificial Intelligence* . 1986. Forthcoming.

[6] Hinrichs, E. Temporale Anaphora im Englischen. 1981.Unpublished ms., University of Tuebingen.

[7] Montague, R. Pragmatics and Intensional Logic. *Synthese* 22:68-94, 1970.

[8] Moser, M.G. An Overview of NIKL, the New Implementation of KL-ONE. In Sidner, C. L., et al. (editors), *Research in Knowledge Representation for Natural Language Understanding - Annual Report, 1 September 1982 - 31 August 1983*, pages 7-26. BBN Laboratories Report No. 5421, 1983.

[9] Polanyi, L. and Scha, R. A Syntactic Approach to Discourse Semantics. In *Proceedings of Int'l. Conference on Computational Linguistics*, pages 413-419. Stanford University, Stanford, CA, 1984.

[10] Scha, R. Semantic Types in PHLIQA1. In *Proceedings of 6th Int'l. Conference on Computational Linguistics*. Ottawa, Canada, 1976.

[11] Scha, R.J.H. Distributive, Collective and Cumulative Quantification. *Formal Methods in the Study of Language, Part 2*. Mathematisch Centrum, Amsterdam, 1981, pages 483-512. Reprinted in: J.A.G. Groenendijk, T.M.V. Janssen and M.B.J. Stokhof (editors). *Truth, Interpretation and Information*. GRASS 3. Dordrecht, Foris, 1984.

[12] Sidner, C.L. Plan parsing for intended response recognition in discourse. *Computational Intelligence* 1(1):1-10, February, 1985.

[13] Stallard, D.G. A Terminological Simplification Transformation for Natural Language Question-Answering Systems. In *Proceedings of the 24th Annual Meeting of the Association for Computational Linguistics*, pages 241-246. Association for Computational Linguistics, June, 1986.

[14] Vilain, M. B. The Restricted Language Architecture of a Hybrid Representation System. In *Proceedings of the Ninth International Joint Conference on Artificial Intelligence (IJCAI-85)*, pages 547-551. The International Joint Conference on Artificial Intelligence, Los Angeles, CA., August, 1985.

2. Out of the Laboratory: A Case Study with the IRUS Natural Language Interface

Ralph M. Weischedel, Edward Walker, Damaris Ayuso, Jos de Bruin,
Kimberle Koile, Lance Ramshaw, Varda Shaked

2.1 Introduction

DARPA's Strategic Computing Program in the application area of Navy Battle Management has provided us several challenges and opportunities in natural language processing research and development. At the beginning of the effort, a set of domain-independent software components, developed through fundamental research efforts dating back as much as seven years, existed. The IRUS software [1] consists of two subsystems: one for linguistic processing and one for adding specifics of the back end. The first subsystem is linguistic in nature, while the second subsystem is not. Linguistic processing includes morphological, syntactic, semantic, and discourse analysis to generate a formula in logic corresponding to the meaning of an English input. The linguistic subsystem is application-independent and also independent of data base interfaces. (This is achieved by factoring all application specifics into the back end processor or into knowledge bases such as dictionary entries and case frame rules, that are domain-specific.) The non-linguistic components convert the logical form to the code necessary for a given underlying system, such as a relational data base.

The IRUS system, or its components, had been used extensively in the laboratory, not just at BBN, but also in research projects at USC/Information Sciences Institute, the University of Delaware, GTE Research, and General Motors Research. However, it had not been exercised thoroughly outside of a research environment.

Our goals in participating in the Strategic Computing Program are manifold:

- To test the collection of state-of-the-art heuristics for natural language processing with a user community trying to solve their problems on a daily basis.
- To test the heuristics on a broad, extensive domain.
- To incorporate research ideas (which are often developed in relative isolation in the laboratory) into a complete system so that effective evaluation and refinement can occur.
- To continue the feedback loop of incorporating new research ideas, testing them in a complete system with real users, evaluating the results, and refining the research accordingly on a repeated basis for several years.

There are several accomplishments in the first year and a half of this work. First, the IRUS software has been delivered to the Naval Ocean Systems Center (NOSC) so that their team may encode the dictionary information, case frame rules, and transformation rules for generating queries appropriate for the underlying systems. The NOSC staff involves a linguist plus individuals trained in computer science, but does not involve experts in natural language processing nor in artificial intelligence. Second, the natural language interface software has been delivered to Texas Instruments (TI), which has integrated it into the Force Requirements Expert System (FRESH). Demonstrations of the natural language interface are being given at several conferences this year as well as to the navy personnel at the Pacific Fleet Command Center. Testing and evaluation of IRUS, both its software and the knowledge bases defined by NOSC for the FCCBMP, will be carried out in the spring of 1986, by the Navy Personnel Research and Development Center.

In this section and section two we present evidence that this is one of the most ambitious applications and tests of natural language processing ever attempted. Section two provides more background regarding the technical challenges inherent in the application environment and in the goals of the Strategic Computing Program. Section three describes what was changed in each system component to support the technology transfer. Section four presents and illustrates the principles that have been underscored in moving this substantial AI system from the laboratory to use: while some principles may appear like common sense, reporting on all the experience should be valuable to future efforts. Section five briefly discusses possible future directions, while section six states our conclusions.

2.2 Background Constraints and Goals

The following sections summarize several constraints and goals which have made this not only a demanding challenge for natural language processing but also an ambitious demonstration of the fruit of AI research.

2.2.1 Multiple Underlying Systems

The decision support environment of the Fleet Command Center Battle Management Program (FCCBMP) involves a suite of decision-making tools. A substantial data base is at the core of those tools and includes roughly 40 relations and 250 fields. In addition, application programs for drawing and displaying maps, various calculations and additional decision support capabilities are provided in the Operations Support Group Prototype (OSGP). In a parallel part of the Strategic Computing Program, two expert systems are being provided: the Force Requirements Expert System (FRESH) and the Capabilities Assessment Expert System (CASES). TI is building the FRESH expert system; the contract for the CASES expert system has not been awarded as of the writing of this paper.

The target users are navy commanders involved in decision making at the Pacific Fleet Command Center; these are top-level executives whose energy is best spent on navy problems and decision making rather than on the details of which of four underlying systems offers a given information capability, on how to divide a problem into the various information capabilities required and how to synthesize the results into the desired answer. Currently they do not access the data base or OSGP application programs themselves; rather, on a round-the-clock basis, two operators are available as intermediates between commander and computer. Consequently, the need for a natural language interface (NLI) is paramount.

2.2.2 The Need For Transportability

There are three ways that transportability has been absolutely required for the natural language interface. First, since we had no experience previously with this application domain, and since the schedule for demonstrations and delivery was highly ambitious, only the application-independent software could be brought to bear on the problem initially; therefore, transportability across application domains was required. Second, the underlying systems have been and will continue to be evolving. For instance, the data base structure is being modified both to support additional information needs for the new expert systems and to provide shorter response time in service of human requests and expert system requests to the data base.

Third, the target output of the natural language interface is subject to change. For instance, the capabilities of FRESH are being developed in parallel with the natural language interface and the CASES expert system has not been started as of this date. Interestingly enough, the target language for the data base could change as well. For instance, there is the possibility of replacing the ORACLE data base management system with a data base machine, in which case the target language would change though the application and data base structure remained constant during the period of installing the data base machine.

2.2.3 Technology Testbed

The project has two goals which at first seem to conflict. First, the software must be hardened enough to be an aid in the daily operations of the Fleet Command Center. Second, the delivered systems are to be a testbed for research results; feedback from use of the systems is to provide a solid empirical base for suggesting new areas of research and refinement of existing research.

As a consequence, software engineering demands placed upon the AI software are quite rigorous. The architecture of the software must support high quality, well worked out, non-toy systems. The software must also support substantial evolution in the heuristics and methods employed as natural language processing provides new research ideas that can be incorporated.

2.3 Adequacy of the Components

In this section we present a brief analysis of the adequacy of the various components in the system, given that the software had not been built with this domain in mind (but had been built with transportability in mind) and given that one of the goals of the effort is to provide a flexible technological base allowing evolution of the techniques and heuristics employed.

2.3.1 Knowledge Representation

At the start of the project, the underlying knowledge representation consisted of a hierarchy of concepts (unary predicates), a list of functions on instances of those concepts, and a list of n-ary predicates. The knowledge representation served several purposes:

- To identify the predicate symbols and function symbols that could be used in the first order logic representing the meaning of sentences,
- To validate selection restrictions (case frame constraints) during the parsing process.

Early on we concluded that greater inference capabilities were required. We wanted to be able to:

- State and reason about knowledge of binary relationships. For instance, every vessel has an arbitrary number of overall readiness ratings associated with it, corresponding to the history of its readiness.
- Represent events and states of affairs flexibly. There may be a variable number of arguments expressed in the input for a given event. For instance, *Admiral Foley deployed the Eisenhower yesterday* or *Admiral Foley deployed the Eisenhower C3.*[2] Also, we needed to be able to count occurrences of events or states of affairs over history, as in *How many times was the the Eisenhower C3 in the last 12 months?* Consequently, we have chosen to represent events and states of affairs as entities, which participate in a number of binary relationships, for instance, specifying the agent, time, location, etc. of the event.

Therefore, the initial ad hoc knowledge representation formalism was replaced with a more general framework, NIKL [10], the new implementation of KL-ONE. This met the needs stated above, and also provided inference mechanisms [15] which could serve as a partial consistency checker on the axioms for the navy domain. Of course, there are other ways to achieve the goals above. However, NIKL was available, and this would be its first use in a technology transfer effort, providing us the opportunity to further explore the power and limitations of limited inference systems.

In NIKL, one can state the classes of entities, the binary relations between entities (including functional relationships), subclass relationships, and subsumption relations among binary relations. It is now used to support:

- The validation of selection restrictions during the parsing process,
- Proposal of possible case frame constraints and possible predicates by the semantic knowledge acquisition component,

[2] *C3* is an overall readiness rating.

- Proposal of the meaning of vague relationships, such as "have", and
- The mapping from first-order logic to relational data base queries.

Once the more powerful knowledge representation and inference mechanisms [15] were available to IRUS, we began using them in unanticipated ways, for instance, the last three in the list above.

2.3.2 The Lexicon and Grammar

The current grammar (RUS) [2] and lexicon are based on the ATN formalism [23]. Though RUS was designed to be a general grammar of dialogue and was clearly among a handful of implemented grammars having the broadest coverage of English, the question was how much modification would be needed for the Navy domain, which was totally new to us.

Very few changes were needed to the software that supports the lexicon and morphological analysis. Those that were required centered around special military forms, such as allowing *06Mar86* as a date and *0600z* as a time. Special symbols and codes such as those are bound to arise in many applications, no matter how transportable the software is.

Very few modifications to the grammar had to be made: those that have been made thus far correspond to special forms and have required very little effort to add. Examples include military (and European) versions of dates, such as *6 March 1986*. This is not to claim that everything a navy user types will be parsed: fully general treatments for conjunction, gapping, and ellipsis, are still research issues for us, as for everyone else. Rather, the experience testifies to the fact that domain-independent grammars can be written for natural language interfaces and that modification of them for a new application can be very small. Sager [12] has reported that few rules of the Linguistic String Parser need to be changed when it is moved to a new application.

The current system handles several classes of ill-formed input, including typographical errors that result in an unknown word: omitted words such as determiners and prepositions; various grammatical errors such as subject verb disagreement and determiner noun disagreement: case errors in using pronouns; and elliptical inputs. The strategy is that of [21].

2.3.3 Semantic Interpretation

Though the software for the semantic interpreter did not depend on domain specifics, the limitations of the initial knowledge representation formalism and of the class of linguistic expressions for which it could compute a semantic representation meant that the semantic interpreter had to be substantially changed. First, the semantic interpreter was modified to take advantage of the stronger knowledge representation formalism and inference available in NIKL. For instance, the interpreter must compute the semantic representation for descriptions of events and states of affairs. It now finds the interpretation of *X has Y* by looking for a relation in the knowledge representation between X and Y.

Second, the semantic interpreter has been changed to correspond more and more to general linguistic analysis. One strength of the initial version of the semantic interpreter [1] was its ability to handle idiomatic expressions, such as *blue forces*. *Blue forces* refers to U.S. forces, as opposed to forces that are blue (in color). The semantic interpreter has been generalized now so that it is much easier to capture the general meaning of *blue* as a predicate, as well as allowing for specification of idiomatic expressions, such as *blue forces*.

A major focus in the next year will be continuing modification of the semantic interpreter so that we have a fully compositional semantics and an intensional logic, rather than a first order logic as the meaning representation of a given sentence. The compositional semantics will still allow, of course, for idiomatic expressions. The enhanced semantic interpreter will be applicable to a much broader class of English expressions, while still being domain-independent and driven by domain-specific case frame rules.

The semantic interpreter does not allow for semantic ill-formedness at present; removing this restriction is a high priority research area.

2.3.4 Discourse Phenomena

Since discourse analysis is the least understood area in natural language processing, the discourse processing component in the system is limited. The system handles anaphora based on the class of the entity required by the selection restrictions upon the anaphor. A benefit of the change in representation making events and states of affairs entities is that the simple heuristic above allows the anaphor in each of the following sequences to be correctly understood:

- *The Eisenhower was deployed C2. When did that occur?*
- *The Eisenhower had been C3. When was that?*

Elliptical inputs that are noun phrases or prepositional phrases are handled as follows: If the class of the entity inherent in the elliptical input is consistent with a class in the previous input, the semantic representation of the new entity is substituted for the semantic representation in the previous input. If not, the ellipsis is interpreted as a request to display the appropriate information.

Far more sophisticated discourse processing is a high priority not only for our project but for natural language work altogether.

2.3.5 Introducing Back end Specifics

The result of linguistic processing in IRUS is a formula in logic. Another component translates the logical expression representing the meaning of an input into an expression in an abstract relational algebra. Simple optimization of the resulting expression is performed in the same component. The initial version of that component (MRLtoERL) [17] used local transformations to translate the n-ary predicates of the logic into the appropriate sequence of projections, joins, etc. on files and fields of the data base.

A straightforward, syntax-directed code generator translates the abstract relational expression into the query language required by the underlying data base management system. Code generators have been built for System 1022, the Britton-Lee Data Base Machine, and ORACLE. An experienced person needs only two to three weeks to create the code generator.

With the move to NIKL and the representation of events and states of affairs as concepts participating in binary relations, the context-free translation of predicates to expressions in relational algebra was no longer adequate. However, the limited inference mechanism [15] of NIKL formed a basis for a simplifier [18] as a preprocess to the MRLtoERL component so that the translation from logic to relational algebra could still be done using only local transformations. Furthermore, the simplifier enabled general translation of linguistic expressions whose data base structure bears little resemblance to the conceptual structure of the English query [18]. We believe the simplification techniques can be generalized further to support the simplification of a subclass of expressions in the intensional logic to be generated by the planned semantic interpreter [19].

Introduction of back end specifics for the OSGP application package and the FRESH expert system is handled by an ad hoc translator from logic to target code at present.

2.3.6 Linguistic Knowledge Acquisition

IRUS's four knowledge bases are:

- The lexicon, which states syntactic and morphological information,
- The taxonomy of case frame rules,
- The model of predicates in the domain, stated in NIKL, and
- The transformation rules for mapping predicates in the logic into projections, joins, etc. of fields in the data base.

The first two of these are linguistic knowledge bases; sophisticated acquisition tools are available to aid the system builder, though not necessarily trained in AI, to build the necessary linguistic knowledge about the vocabulary.

Powerful knowledge acquisition tools for building these domain-specific constraints could greatly ease the process of bringing up a natural language interface for a new application and consequently for broadening the applicability of NLI technology. Perhaps the most powerful demonstration of acquisition tools to date has been

TEAM [6]. Based on the fields and files of a given data base, TEAM's acquisition tools lead the individual through a sequence of questions to acquire the specific linguistic and domain knowledge needed to understand a broad subset of language for querying the data base. However, since those heuristics are in large part specific to the task of accessing data bases, that technology could not be directly applied to the FCCBMP application, which encompasses a relational data base, an application package including both map drawing and calculation, and expert systems.

Knowledge acquisition tools for IRUS, developed under earlier DARPA-funded work at BBN, were not specific to data base applications and therefore could be applied in the FCCBMP. Even if applicability of the TEAM heuristics were not a problem, there are theoretical and technical difficulties in translating English requests into data base queries [9] which would argue for a more general approach such as ours. As Scha [13, 14] has argued, these difficulties, as well as the issues of transportability and generality, suggest keeping linguistic knowledge rather independent of assumptions about the back end.

IRACQ, the semantic acquisition tool made available to NOSC for specifying case frames and their associated translations, is quite powerful. The initial version [11] allowed one to specify the case frame for a new word sense by giving an example of a phrase using that word sense. For instance, if *the admiral, a vessel,* and *C2* are known to the system, then one can define a new case frame for *deploy* by giving a phrase such as *the admiral deployed a vessel C2*. The system suggests generalizations of the arguments specified in the example using the NIKL knowledge base, so that the inferred case frame is the most general that the user authorizes. For example, generalizations of admiral are commanding officer, person, and physical object; generalizations of vessel are unit, platform, and physical object; generalizations of C2 are rating and code. Furthermore, based on the introduction of the more general knowledge representation system NIKL, IRACQ is being extended to propose the binary relations that might be part of the translation of the new word. Of course, if the relations and concepts needed are not already present in the domain predicate model, the user can define new concepts and relations in the NIKL hierarchy as well.

The availability of such knowledge acquisition tools has made it possible for NOSC representatives, rather than AI experts, to define the naval language expected as input. We have found that even with the tool described above, reasonable linguistic sophistication is very helpful in defining the case frames. In fact, an individual with a master's degree in linguistics is defining the case frames at NOSC More sophisticated tools, which do not presuppose only one kind of back end, are one of the most important research topics for natural language interfaces. These would combine the strengths of the linguistic knowledge acquisition tools of both IRUS and TEAM.

2.4 Principles Underscored

In the course of the effort, a number of principles have been underscored. Many of these once stated may appear to be common sense; however, we hope that illustrating them from our experience will prove helpful.

2.4.1 The Necessity For General Solutions

The availability of domain-independent software driven by domain-dependent, declarative knowledge bases was of paramount importance because of the following:

- The application was not only broad (three underlying systems) but also evolving (with a fourth system to be added).
- Great habitability is necessary for delivery to the Pacific Fleet Command Center.
- The time frame for demonstration was relatively short compared to the scope of the underlying systems to be covered.

Furthermore, it is critical that the knowledge bases state a linguistic or domain fact once and that the domain-independent software be able to use that one fact in all predictable linguistic variations. The reasons are obvious: the efficiency in building the knowledge bases, the consistency of stating a fact only once, and the habitability of the resulting system which can understand things no matter what form they are expressed in.[3]

The IRUS system attains the goal mentioned above relatively well; a linguistic or application constraint is stated once in the knowledge base but applied in all possible ways in the language processing. This is particularly true because of the substantial grammar [2, 3] and to a lesser extent due to the semantic interpreter. Recognition of this fact is part of the reason that substantial changes, as mentioned in section three, are planned in the semantic interpreter to make the linguistic facts that drive it even more general.

[3] An interesting anecdote that arose in early discussions in the planning of this project centered around the tight deadlines and the breadth of the application area. Since it was clear that one could not cover all three underlying systems in every area for which they could provide information, the question arose whether to focus on a substantial subpart of the application domain initially or to sacrifice linguistic coverage to gain in coverage of the underlying systems. Because the information needs of the various navy personnel differed widely, and because the scope of needs seemed impossible to predict, navy personnel initially suggested that coverage of all possible information stored in the underlying systems was of such importance that sacrifices regarding the language understood could be made even if there were only one way that a given piece of information could be accessed. The interesting thing however is that as demonstrations were given, the first things people request following the demonstration is to try various rephrasings of the requests in the demonstration, thereby in behavior indicating how important not being restricted to special forms is.

2.4.2 The Necessity of Heuristic Solutions

In the previous section we have argued for the need of general purpose solutions to problems in NLI. Clearly this cannot be taken to an extreme; otherwise one would not have an NLI in the foreseeable future, since there are well-known outstanding problems for which there is no general, comprehensive solution on the horizon. Consequently, heuristic, state-of-the-art solutions are being demonstrated for problems such as ambiguity, vagueness, discourse context, ill-formed input, definite reference, quantifier scope, conjunction, and ellipsis. Though laboratory use of the system embodying that set of heuristics is quite promising, we expect that placing the system in the hands of individuals trying to solve their day-to-day problems will produce interesting corpora of dialogues that cannot be handled by one or more of those heuristics. Careful study of those corpora will tell us not only the effectiveness of state-of-the-art solutions but will also suggest new directions of research.

2.4.3 The Necessity of Extra-linguistic Elements in a Natural Language Interface

Having only a natural language processor is not sufficient to provide a truly natural interface. Four elements seem highly valuable for typed input: editing, a readily accessible history of the session, human factors elements in the presentation, and a minimum of key strokes. Editing should include more than deleting the last character of the string and deleting the whole string. We are currently relying on Emacs, which is readily available on Symbolics workstations. However, that is also unattractive because of the arcane nature of the link between the myriad control key commands of Emacs and the actual textual tasks the user needs to perform.

IRUS's on-line history of the session provides reviewing earlier results, editing the text of earlier requests to create new ones, and generating a standard protocol for routine operations that occur on a regular basis. Our user community anticipates a need for both routine sequences of questions as would be useful in preparing daily or weekly reports, and ad hoc queries, e.g., when crises arise.

Issues in presentation are important as well. No matter what the underlying application is, IRUS lets it produce output on the complete bitmap screen. A popup input window and an optional popup history window can be moved to any part of the screen so that all parts of the underlying system's output may be visible.

Certain operations occur so frequently that one would like to have them available on the screen at all times in menus to minimize memory load and key strokes. Examples are clearing a window and aborting a request.

A future capability that would be quite attractive is pointing to individual data items, classes of data items, field headings, or locations on maps, causing the appropriate linguistic description of that entity to be made available as part of the natural language input. While this is possible in the future, providing such a capability is not currently funded.

Speech input as a mode of communication would also be highly desirable, even if extremely limited initially. As a consequence, the next generation of natural language understanding systems in the FCCBMP will include modifications specifically to provide an infrastructure which could at a later date support speech input.

2.5 Future Possibilities

In addition to the enhancements we have mentioned earlier regarding the semantic interpreter, linguistic knowledge acquisition tools, and discourse processing, there are three substantial areas of research and development possible. First, research in ill-formed input is necessary in order to allow for additional grammatical problems in the input and for relaxation of semantic constraints, e.g., to allow for figures of speech. The problem with an ill-formed input is that there is no interpretation which satisfies all linguistic constraints. Therefore, the very constraints that limit search must be relaxed, thereby opening Pandora's Box in terms of the number of alternatives in the search space. Not only IRUS, but apparently all systems that process any ill-formed input attain the success they do by considering very few kinds of ill-formed input and by assuming that semantic constraints can never be violated.[4] Consequently, determining what the user meant in an ill-formed input is a substantial problem requiring research.

Second, we propose exploring parallel architectures to add functional capability. Run time performance of IRUS on a Symbolics machine is quite acceptable. Typical inputs are fully processed to give the target language input to the underlying system within a few seconds; naturally, the relational data base and underlying expert systems are not expected to be able to perform at comparable speeds. There are three areas where functional performance could be improved by parallelism.

1. The current system ranks the partial parses using both semantic and syntactic information, and it explores those partial parses based on following up the most promising one first. The technique is relatively effective, but clearly not infallible. Finding all interpretations and then ranking them based not only on local syntactic and semantic tests but also on global semantic, pragmatic, and discourse information is critical to improving the identification of what the user intended.
2. A second area related to the first, is greater coverage of ill-formed input. As mentioned earlier, ill-formedness requires relaxing the rules that constrain search; therefore the search space grows dramatically in processing an ill-formed input.
3. Real-time, large vocabulary, large branching factor, continuous speech recognition is beyond the state of the art, and requires highly parallel machines to support speech signal processing. While this is highly desirable, it is not part of our current effort.

Within the next two years we intend to replace the ATN grammar with a declarative, side-effect free grammar and a parallel parsing algorithm, following work reported in [16].

Third, our evolving system is being interfaced to the Penman generation component from USC/Information Sciences Institute (USC/ISI) [8]. Penman is based upon systemic linguistics. The ultimate goal of the effort with USC/ISI is twofold: to have systems that can understand whatever they generate and to achieve this by having common knowledge sources for the lexicon, for the NIKL model of domain predicates, and for discourse information.

[4] Early work on allowing semantic relaxation is reported in [5, 21, 22].

2.6 Conclusions

Though the project will be ongoing for several years yet, there are several preliminary conclusions from the first year and a half of effort, given the constraints and goals mentioned in section two.

1. Providing language coverage for this broad application with multiple underlying systems has not been a problem. However, since determining what system(s) must be accessed for a given input is a research problem that has been little addressed, only simple linguistic clues are used in the current version. The problem in general involves not only reasoning about the capabilities of the underlying systems [7] but also significant linguistic issues. For instance, if one says *Show me the carriers whose condition code changed in the last 24 hours*, either a list (from the data base) or a map (from OSGP) is appropriate. If one says *Show me a display of the carriers whose condition code changed in the last 24 hours*, only OSGP is appropriate. The linguistic cue is *display*. Furthermore, some contexts favor one over the other.

2. The architecture has supported transportability well. For instance, this new application required only minor changes to the grammar and morphological analyzer. As FRESH has been further defined and as the data base structure has evolved, only small local changes have been required to the content of the knowledge bases. Should a data base machine replace the current data base management system in Hawaii, only two to three person weeks should be needed to generate the new target language. However, more sophisticated linguistic knowledge acquisition tools not dependent on the type of the underlying application system are a critical goal for NLI both for far greater applicability of the technology and for far broader availability of NLIs.

3. The success of this effort as a technology testbed depends on evaluation after installation at the Pacific Fleet Command Center and on the success of the architecture to support substantial enhancements, such as the planned semantic interpreter based on compositional semantics and the planned parallel parser. However, it already has supported massive changes well, such as the change in underlying knowledge representation when NIKL was introduced.

   The potential of the testbed is great because it offers empirical research of a realistic kind unfortunately largely lacking heretofore; the placement of TQA in the hands of users to solve their daily problems for a year [4] is a notable exception. The results of research on heuristics for definite reference; semantic ambiguity; ellipsis; syntactically or semantically ill-formed input; and inference from world knowledge and context, to name a few studied in isolation, must be tested in a complete system. The opportunity in the FCCBMP will help to determine the effectiveness of such heuristics in a large diverse application domain where combinatorial issues cannot be ignored. Collecting corpora in an experiment can be highly instructive, as shown in [20]. However, corpus collection using people solving their own problems provides an uncommon degree of realism and legitimacy to the empirical process.

References

[1] Bates, M., Stallard, D., and Moser, M. The IRUS Transportable Natural Language Database Interface. *Expert Database Systems*. Cummings Publishing Company, Menlo Park, CA. 1985.

[2] Bobrow, R.J. *The RUS System*. BBN Report 3878, Bolt Beranek and Newman Inc., 1978.

[3] Bobrow, R. and Bates, M. The RUS Parser Control Structure. In *Research in Knowledge Representation for Natural Language Understanding, Annual Report*. Bolt Beranek and Newman Inc., 1982. BBN Report No. 5188.

[4] Damerau, F.J. Operating Statistics for the Transformational Question Answering System. *American Journal of Computational Linguistics* 7(1):30-42, 1981.

[5] Fass, D. and Wilks, Y. Preference Semantics, Ill-Formedness, and Metaphor. *American Journal of Computational Linguistics* 9(3-4):178-187, 1983.

[6] Grosz, B., Appelt, D. E., Martin, P., and Pereira, F. *TEAM: An Experiment in the Design of Transportable Natural Language Interfaces*. Technical Report 356, SRI International, 1985. To appear in Artificial Intelligence.

[7] Kaczmarek, T., Mark, W., and Sondheimer, N. The Consul/CUE Interface: An Integrated Interactive Environment. In *Proceedings of CHI '83 Human Factors in Computing Systems*, pages 98-102. ACM, December, 1983.

[8] Mann, W.C. and Matthiessen, C.M.I.M. Nigel: A Systemic Grammar for Text Generation. *Systemic Perspectives on Discourses: Selected Theoretical Papers from the 9th International Systemic Workshop*. Ablex, Norwood, NJ, forthcoming.

[9] Moore, R.C. Natural Language Access to Databases - Theoretical/Technical Issues. In *Proceedings of the 20th Annual Meeting of the Association for Computational Linguistics*, pages 44-45. Association for Computational Linguistics, June, 1982.

[10] Moser, M.G. An Overview of NIKL, the New Implementation of KL-ONE. In Sidner, C. L., et al. (editors), *Research in Knowledge Representation for Natural Language Understanding - Annual Report, 1 September 1982 - 31 August 1983*, pages 7-26. BBN Laboratories Report No. 5421, 1983.

[11] Moser, M.G. Domain Dependent Semantic Acquisition. In *The First Conference on Artificial Intelligence Applications*, pages 13-18. IEEE Computer Society, December, 1984.

[12] Sager, N. The String Parser for Scientific Literature. In R. Rustin (editor), *Natural Language Processing*, pages 61-88. Algorithmics Press, Inc., New York, NY, 1973.

[13] Scha, R.J.H. English Words and Data Bases: How to Bridge the Gap. In *Proceedings of the 20th Annual Meeting of the Association for Computational Linguistics*, pages 57-59. Association for Computational Linguistics, June, 1982.

[14] Scha, R.J.H. *Logical Foundations for Question Answering*. Technical Report, Eindhoven: Philips Research Labs, M.S. 12.331., 1983.

[15] Schmolze, J.G., Lipkis, T.A. Classification in the KL-ONE Knowledge Representation System. In *Proceedings of the Eighth International Joint Conference on Artificial Intelligence*. 1983.

[16] Sridharan, N.S. *Semi-Applicative Programming: Examples of Context Free Recognizers*. Technical Report Report No. 6135, BBN Laboratories Inc., January, 1986.

[17] Stallard, D. Data Modelling for Natural Language Access. In *The First Conference on Artificial Intelligence Applications*, pages 19-24. IEEE Computer Society, December, 1984.

[18] Stallard, D.G. A Terminological Simplification Transformation for Natural Language Question-Answering Systems. In *Proceedings of the 24th Annual Meeting of the Association for Computational Linguistics*, pages 241-246. Association for Computational Linguistics, June, 1986.

[19] Stallard, D.G. *Taxonomic Inference on Predicate Calculus Expressions.* Technical Report, BBN Laboratories Inc., 1986. In preparation.

[20] Thompson, B.H. Linguistic Analysis of Natural Language Communication with Computers. In *Proceedings of the Eighth International Conference on Computational Linguistics*, pages 190-201. International Committee on Computational Linguistics, October, 1980.

[21] Weischedel, R. M. and Sondheimer, N. K. Meta-rules as a Basis for Processing Ill-Formed Input. *American Journal of Computational Linguistics* 9(3-4):161-177, 1983.

[22] Weischedel, R.M. and Sondheimer, N.K. *Relaxing Constraints in MIFIKL.* Technical Report, USC/Information Sciences Institute, 1983.

[23] Woods, W.A. Transition Network Grammars for Natural Language Analysis. *CACM* 13(10):591-606, October, 1970.

3. A Terminological Simplification Transformation for Natural Language Question-Answering Systems[5]

David G. Stallard

3.1 Introduction

A common and useful strategy for constructing natural language interface systems is to divide the processing of an utterance into two major stages: the first mapping the utterance to a logical expression representing its "meaning" and the second producing from this logical expression the appropriate response. The second stage is not necessarily trivial: the difficulty of its design is significantly affected by the complexity and generalness of the logical expressions it has to deal with. If this issue is not faced squarely, it may affect choices made elsewhere in the system. Indeed, a need to restrict the form of the meaning representation can be at odds with particular approaches towards producing it - as for example the "compositional" approach, which does not seek to control expression complexity by giving interpretations for whole phrasal patterns, but simply combines together the meaning of individual words in a manner appropriate to the syntax of the utterance. Such a conflict is certainly not desirable: we want to have freedom of linguistic action as well as to be able to obtain correct responses to utterances.

This paper treats in detail the particular manifestation of these issues for natural-language systems which serve as interfaces to a database: the problems that arise in a module which maps the meaning representation to a second logical language for expressing actual database queries. A module performing such a mapping is a component of such question-answering systems as TEAM [4], PHLIQA1 [8] and IRUS [1]. As an example of difficulties which may be encountered, consider the question "Was the patient's mother a diabetic?" whose logical representation must be mapped onto a particular boolean field which encodes for each patient whether or not this complex property is true. Any variation on this question which a compositional semantics might also handle, such as "Was diabetes a disease the patient's mother suffered from?", would result in a semantically equivalent but very different-looking logical expression: this different expression would also have to be mapped to this field. How to deal with these and many other possible variants, without making the mapping process excessively complex, is clearly a problem.

The solution which this paper presents is a new level of processing, intermediate between the other two: a novel simplification transformation which is performed on the result of semantic interpretation before the attempt is

[5] This paper was originally published in the *Proceedings of the 24th Annual Meeting of the Association for Computational Linguistics*, 10-13 June, 1986, Columbia University, New York. Requests for copies should be addressed to: Dr.Donald E. Walker (ACL), Bell Communications, Research, 435 South Street MRE 2A379, Morristown, NJ 07960, USA

made to map it to the database. This simplification method relies on knowledge which is stored in a taxonomic knowledge representation system such as NIKL [5]. The principle behind the method is that an expression may be *simplified by translating its subexpressions, where possible, into the language of NIKL, and classifying the result into the taxonomy to obtain a simpler equivalent for them.* The result is to produce an equivalent but syntactically simpler expression in which fewer, but more specific, properties and relations appear. The benefit is that deductions from the expression may be more easily "read off"; in particular, the mapping becomes easier because the properties and relations appearing are more likely to be either those of the database or composable from them.

The body of the paper is divided into four sections. In the first, I will summarize some past treatments of the mapping between the meaning representation and the query language, and show the problems they fail to solve. The second section prepares the way by showing how to connect the taxonomic knowledge representation system to a logical language used for meaning representation. The third section presents the "recursive terminological simplification" algorithm itself. The last section describes the implementation status and suggests directions for interesting future work.

3.2 A Formal Treatment of the Mapping Problem

This section discusses some previous work on the problem of mapping between the logical language used for meaning representation and the logical language in which actual database queries are expressed. The difficulties which remain for these approaches will be pointed out.

A common organization for a database is in terms of tables with rows and columns. The standard formulation of these ideas is found in the relational model of Codd [3], in which the tables are characterized as relations over sets of atomic data values. The elements (rows) of a relation are called "tuples", while its individual argument places (columns) are termed its "attributes". Logical languages for the construction of queries, such as Codd's relational algebra, must make reference to the relations and attributes of the database.

The first issue to be faced in consideration of the mapping problem is what elements of the database to identify with the objects of discourse in the utterance - that is, with the non-logical constants[6]in the meaning representation. In previous work [9] I have argued that these should not be the rows of the tables, as one might first think, but rather certain sets of the atomic attribute-values themselves. I presented an algorithm which converted expressions of a predicate calculus-based meaning representation language to the query language ERL, a relational algebra [3] extended with second-order operations. The translations of non-logical constants in the meaning representation were provided by fixed and local translation rules that were simply ERL expressions for computing the total extension of the constant in the database. The expressions so derived were then combined together in an

[6] This term, while a standard one in formal logic, may be confused with other uses of the word "constant". It simply refers to the function, predicate and ordinary constant symbols, such as "MOTHER" or "JOHN", whose denotations depend on the interpretation of the language, as opposed to fixed symbols like "FORALL","AND", "TRUE".

appropriate way to yield an expression for computing the response for the entire meaning representation expression. If the algorithm encountered a non-logical constant for which no translation rule existed, the translation failed and the user was informed as to why the system could not answer his question.

By way of illustration, consider the following relational database, consisting of clinical history information about patients at a given hospital and of information about doctors working there:

```
PATIENTS(PATID, SEX, AGE, DISEASE, PHYS, DIAMOTHER)
DOCTORS(DOCID, NAME, SEX, SPECIALTY)
```

where "PHYS" is the ID of the treating physician, and "DIAMOTHER" is a boolean field indicating whether or not the patient's mother is diabetic. Here are the rules for the one-place predicate PATIENTS, the one-place predicate SPECIALTIES, and the two-place predicate TREATING-PHYSICIAN:

```
 PATIENTS    => (PROJECT PATIENTS OVER PATID)

SPECIALTIES    => (PROJECT DOCTORS OVER SPECIALTY)

TREATING-PHYSICIAN    => (PROJECT (JOIN PATIENTS
                                       TO DOCTORS
                                       OVER PHYS DOCID)
                                 OVER PATID DOCID)
```

Note that while no table exists for physician SPECIALTIES, we can nonetheless give a rule for this predicate in way that is uniform with the rule given for the predicate PATIENTS.

One advantage of such local translation rules is their simplicity. Another advantage is that they enable us to treat database question-answering model-theoretically. The set-theoretic structure of the model is that which would be obtained by generating from the relations of the database the much larger set of "virtual" relations that are expressible as formulas of ERL. The interpretation function of the model is just the translation function itself. Note that it is a *partial* function because of the fact that some non-logical constants may not have translations. We speak therefore of the database constituting a "partially specified model" for the meaning representation language. Computation of a response to a user's request, instead of being characterizable only as a procedural operation, becomes interpretation in such a model.

A similar model-theoretic approach is advocated in the work on PHLIQA1 [7], in which a number of difficulties in writing local rules are identified and overcome. One class of techniques presented there allows for quite complex and general expressions to result from local rule application, to which a post-translation simplification process is applied. Other special-purpose techniques are also presented, such as the creation of "proxies" to stand in for elements of a set for which only the cardinality is known.

A more difficult problem, for which these techniques do not provide a general treatment, arises when we want to get at information corresponding to a complex property whose component properties and relations are not themselves stored. For example, suppose the query "List patients whose mother was a diabetic", is represented by the meaning representation:

```
(display ^(setof X:PATIENT
              (forall Y:PERSON (->(MOTHER X Y)
                                  (DIABETIC Y)))))
```

The information to compute the answer may be found in the field DIAMOTHER above. It is very hard to see how we will use local rules to get to it, however, since nothing constructable from the database corresponds to the non-logical constants MOTHER and DIABETIC. The problem is that the database chooses to highlight the complex property DIAMOTHER while avoiding the cost of storing its constituent predicates MOTHER and DIABETIC - the conceptual units corresponding to the words of the utterance.

One way to get around these difficulties is of course to allow for a more general kind of transformation: a "global rule" which would match against a whole syntactic pattern like the universally quantified sub-expression above. The disadvantage of this, as is pointed out in [7], is that the richness of both natural language and logic allows the same meaning to be expressed in many different ways, which a complete "global rule" would have to match. Strictly syntactic variation is possible: pieces of the pattern may be spread out over the expression, from which the pattern match would have to grab them. Equivalent formulations of the query may also use completely different terms. For example, the user might have employed the equivalent phrase "female parent" in place of the word "mother", presumably causing the semantic interpretation to yield a logical form with the different predicates "PARENT" and "FEMALE". This would not match the pattern. It becomes clear that the "pattern-matching" to be performed here is not the literal kind, and that it involves unspecified and arbitrary amounts of inference.

The alternative approach presented by this paper takes explicit account of the fact that certain properties and relations, like "DIAMOTHER", can be regarded as built up from others. In the next section we will show how the properties and relations whose extensions the database stores can be axiomatized in terms of the ones that are more basic in the application domain. This prepares the way for the simplification transformation itself, which will rely on a limited and sound form of inference to reverse the axiomatization and transform the meaning representation, where possible, to an expression that uses only these database properties and relations. In this way, the local rule paradigm can be substantially restored.

3.3 Knowledge Representation and Question-Answering

The purpose of this section of the paper is to present a way of connecting the meaning representation language to a taxonomic knowledge representation system in such a way that the inference-making capability of the latter is available and useful for the problems this paper addresses. Our approach may be contrasted with that of others, e.g. TEAM in which such a taxonomy is used mainly for simple inheritance and attachment duties.

The knowledge representation system used in this work is NIKL [5]. Since NIKL has been described rather fully in the references, I will give only a brief summary here.

NIKL is a taxonomic frame-like system with two basic data structures: concepts and roles. Concepts are just classes of entities, for which roles function somewhat as attributes. At any given concept we can restrict a role to be filled by some other concept, or place a restriction on the number of individual "fillers" of the role there. A role has one concept as its "domain" and another as its "range": the role is a relation between the sets these two concepts denote. Concepts are arranged in a hierarchy of sub-concepts and superconcepts; roles are similarly arranged. Both concepts and roles may associated with names. In logical terms, a concept may be identified as the one-place predicate with its name, and a role as the two-place predicates with its name.

I will now give the meaning postulates for a term-forming algebra, similar to the one described in [2] in which one can write down the sort of NIKL expressions I will need. Expressions in this language are combinable to yield a complex concept or role as their value.

```
(CONJ C1 -- CN)    ≡   (lambda (X) (and (C1 X) -- (Cn X)))

(VALUERESTRICT R C)  ≡  (lambda (X) (forall Y (-> (R X Y)
                                                   (C Y)))

(NUMBERRESTRICT R 1 NIL) #≡# (lambda (X) (exists Y (R X Y)))

(VRDIFF R C)  ≡  (lambda (X Y) (and (R X Y) (C Y)))

(DOMAINDIFF R C)  ≡ (lambda (X Y) (and (R X Y) (C X)))
```

The key feature of NIKL which we will make use of is its classifier, which computes subsumption and equivalence relations between concepts, and a limited form of this among roles. Subsumption is sound, and thus indicates entailment between terms:

```
    (SUBSUMES C1 C2)   ->   (forall X (-> (C2 X) (C1 X)))
```

If the classifier algorithm is complete, the reverse is also true, and entailment indicates subsumption. Intuitively, this means that classified concepts are pushed down as far in the hierarchy as they can go.

Also associated with the NIKL system, though not a part of the core language definition is a symbol table which associates atomic names with the roles or concepts they denote, and concepts and roles with the names denoting them. If a concept or role does not have a name, the symbol table is able to create and install one for it when demanded.

3.3.1 The Domain Model

In order to be able to use NIKL in the analysis of expressions in the meaning representation language, we make the following stipulations for any use of the language in a given domain. First, any one-place predicate must name a concept, and any two-place predicate name a role. Second, any constant, unless a number or a string, must name an "individual" concept - a particular kind of NIKL concept that is defined to have at most one member. N-ary functions are treated as a N+1 - ary predicates. A predicate of N arguments, where N is greater than 2, is reified as a concept with N roles. This set of concepts and roles, together with the logical relationships between them, we call the "domain model".

Note that all we have done is to stipulate an one-to-one correspondence between two sets of things - the concepts and roles in the domain model and the non-logical constants of the meaning representation language. If we wish to include a new non-logical constant in the language we must enter the corresponding concept or role in the domain model. Similarly, the NIKL system's creating a new concept or role, and creation of a name in the symbol table to stand for it, furnishes us with a new non-logical constant.

3.3.2 Axiomatization of the Database in Terms of the Domain Model

The translation rules presented earlier effectively seek to axiomatize the properties and relations of the domain model in terms of those of the database. This is not the only way to bridge the gap. One might also try the reverse: to axiomatize the properties and relations of the database in terms of those of the domain model. Consider the DIAMOTHER field of our sample database. We can write this in NIKL as the concept PATIENT-WITH-DIABETIC-MOTHER using terms already present in the domain model:

```
(CONJ PATIENT
      (VALUERESTRICT MOTHER
                     DIABETIC))
```

If we wanted to axiomatize the relation implied by the SEX attribute of the PATIENTS table in our database, we could readily do so by defining the role PATIENT-SEX in terms of the domain model relation SEX:

```
(DOMAINDIFF SEX
         PATIENT)
```

These two defined terms can actually be entered into the model, and be treated just like any others there. For example, they can now appear as predicate letters in meaning representations. Moreover, to the associated data structure we can attach a translation rule, just as we have been doing with the original domain model elements. Thus, will attach to the concept PATIENT-WITH-DIABETIC-MOTHER the rule:

```
(PROJECT (SELECT FROM PATIENTS WHERE (EQ DIAMOTHER "YES"))
         OVER PATID)
```

The next section will illustrate how we map from expressions using "original" domain model elements to the ones we create for axiomatizing the database, using the NIKL system and its classifier.

3.4 Recursive Terminological Simplification

We now present the actual simplification method. It is composed of two separate transformations which are applied one after the other. The first, the "contraction phase", seeks to contract complicated subexpressions (particularly nested quantifications) to simpler one-place predications, and to further restrict the "sorts" of remaining bound variables on the basis of the one-place predicates so found. The second part of the transformation, the "role-tightening" phase, replaces general relations in the expression with more specific relations which are lower in the NIKL hierarchy. These more specific relations are obtained from the more general by considering the sorts of the variables upon which a given relational predication is made.

3.4.1 The Contraction Phase

The contraction phase is an algorithm with three steps, which occur sequentially upon application to any expression of the meaning representation. First, the contraction phase applies itself recursively to each non-constant subexpression of the expression. Second, depending upon the syntactic category of the expression, one of the "pre-simplification" transformations is applied to place it in a normalized form. Third and finally, one of the actual simplification transformations is used to convert the expression to one of a simpler syntactic category.

Before working through the example, I will lay out the transformations in detail. In what follows, X and X1,X2 -- Xn are variables in the meaning representation language. The symbol "<rest>" denotes a (possibly empty) sequence of formulae. The expression "(FORMULA X)" denotes a formula of the meaning representation language in which the variable X (and perhaps others) appears freely. The symbol "<quant>" is to be understood as being replaced by either the operator SETOF or the quantifier EXISTS.

First, the normalization transformations, which simply re-arrange the constituents of the expressions to a more convenient form without changing its syntactic category:

```
(1)   (and (P1 X1) (P2 X1) -- (PN X1)
           (Q1 X2) (Q2 X2) -- (QN X2)
           <rest>)

        ==>  (and (P' X1) (Q' X2) <rest>)

       where P'  := (CONJ P1 P2 -- PN)
       and   Q'  := (CONJ Q1 Q2 -- QN)

(2)   (<quant> X:S (and (P X) <rest>)  ==>

       (<quant> X:S' (and <rest>))
       where S'  := (CONJ S P)

(3)  (<quant> X:S (P X))  ==>

       (<quant> X:S')
       where S'  := (CONJ S P)

(4)  (forall X:S (-> (and (P X) <rest>)
                     (FORMULA X))   ==>

       (forall X:S' (-> (and <rest>)
                        (FORMULA X)))
```

In (2) and (4) above, the conjunction or implication, respectively, are collapsed out if the sequence <rest> is empty.

Now the actual simplification transformations, which seek to reduce a complex sub-expression to a one-place predication.

```
(5)  (forall X2:S (-> (R X1 X2) (P X2)))
       ==>     (P' X1)
      where P'  := (VALUERESTRICT (VRDIFF R S) P)
```

```
(6) (exists X2:S (R X1 X2))  ==>  (P' X1)

       where P' := (VALUERESTRICT R S)
       and R must be a functional role

(7) (exists X2:S (R X1 X2))   ==> (P' X1)

      where P  := (NUMBERRESTRICT (VRDIFF R S) 1 NIL)

(8) (and (P X))  ==> (P X)

(9) (R X C) ==> (P X)

       where P := (VALUERESTRICT R C)
       and R is functional, C an individual concept
```

Now let us suppose that the exercise at the end of the last section has been carried out, and that the concept PATIENT-WITH-DIABETIC-MOTHER has been created and given the appropriate translation rule. To return to the query "List patients whose mother was a diabetic", we recall that it has the meaning representation:

```
   (DISPLAY ^(SETOF X:PATIENTS
                    (FORALL Y:PERSON
                            (-> (MOTHER X Y)
                                (DIABETIC Y))))))
```

Upon application to the SETOF expression, the algorithm first applies itself to the inner FORALL. The syntactic patterns of none of the pre-simplification transformations (2) - (4) are satisfied, so transformation (5) is applied right way to produce the NIKL concept:

```
     (VALUERESTRICT (VRDIFF MOTHER PERSON)
                    DIABETIC)
```

This is given to the NIKL classifier, which compares it to other concepts already in the hierarchy. Since MOTHER has PERSON as its range already, (VRDIFF MOTHER PERSON) is just MOTHER again. The classifier thus computes that the concept specified above is a subconcept of PERSON - a PERSON such that his MOTHER was a DIABETIC. If this is not found to be equivalent to any pre-existing concept, the system assigns the concept a new name which no other concept has, say PERSON-1. The outcome of the simplification of the whole FORALL is then just the much simpler expression:

```
                   (PERSON-1 X)
```

The recursive simplification of the arguments to the SETOF is now completed, and the resulting expression is:

```
      (DISPLAY ^(SETOF X:PATIENT
                       (PERSON-1 X)))
```

Transformations can now be applied to the SETOF expression itself. The pre-simplification transformation (3) is found to apply, and a concept expressed by:

```
                   (CONJ PATIENT PERSON-1)
```

is given to the classifier, which recognizes it as equivalent to the already existing concept PATIENT-WITH-DIABETIC-MOTHER. Since any concept can serve as a sort, the final simplification is:

```
 (DISPLAY ^(SETOF X:PATIENT-WITH-DIABETIC-MOTHER))
```

This is the very concept for which we have a rule, so the ERL translation is:

```
(PRINT FROM (SELECT FROM PATIENT
                    WHERE (EQ DIAMOTHER "YES"))
       PATID)
```

Suppose now that the semantic interpretation system assigned a different logical expression to represent the query "List patients whose mother was a diabetic", in which the embedded quantification is existential instead of universal. This might actually be more in line with the number of the embedded noun. The meaning representation would now be:

```
(display ^(setof X:PATIENT
                 (exists Y:PERSON (and (MOTHER X Y)
                                       (DIABETIC Y))))
```

The recursive application of the algorithm proceeds as before. Now, however, the pre-simplification transformation (2) may be applied to yield:

```
(exists Y:DIABETIC (MOTHER X Y))
```

since a DIABETIC is already a PERSON. Transformation (6) can be applied if MOTHER is a "functional" role - mapping each and every person to exactly one mother. This can be checked by asking the NIKL system if a number restriction has been attached at the domain of the role, PERSON, specifying that it have both a minimum and a maximum of one. If the author of the domain model has provided this reasonable and perfectly true fact about motherhood, (6) can proceed to yield:

```
(PATIENT-WITH-DIABETIC-MOTHER X)
```

as in the preceding example.

3.4.2 The Role Tightening Phase

This phase is quite simple. After the contraction phase has been run on the whole expression, a number of variables have had their sorts changed to tighter ones. This transformation sweeps through an expression and changes the roles in the expression on that basis. Thus:

```
(10) (R X Y)  ==>  (R' X Y)

     where S1 is the sort of X
     and S2 is the sort of Y
     and R' := (DOMAINDIFF (VRDIFF R S2)
                           S1)
```

One can see that a use of the relation SEX, where the sort of the first argument is known to be DOCTOR, can readily be converted to a use the relation DOCTOR-SEX.

3.4.3 Back Conversion: Going in the Reverse Direction

There will be times when the simplification transformation will "overshoot", creating and using new predicate letters which have not been seen before by classifying new data structures into the model to correspond to them. The use of such a new predicate letter can then be treated exactly as would its equivalent lambda-definition, which we can readily obtain by consulting the NIKL model. For example, a query about the sexes of leukemia victims may after simplification result in a rather strange role being created and entered into the hierarchy:

```
PATIENT-SEX-1 := (DOMAINDIFF PATIENT-SEX LEUKEMIA-PATIENT)
```

This role is a direct descendant of PATIENT-SEX; its name is system generated. By the meaning-postulate of DOMAINDIFF given in section 3 above, it can be rewritten as the following lambda-abstract:

```
(lambda (X Y) (and (PATIENT-SEX X Y)
                   (LEUKEMIA-PATIENT X)))
```

For PATIENT-SEX we of course have a translation rule as discussed in section 2. A rule for LEUKEMIA-PATIENT can be imagined as involving the DISEASE field of the PATIENTS table. At this point we can simply call the translation algorithm recursively, and it will come up with a translation:

```
(PROJECT (SELECT FROM PATIENTS
                 WHERE (EQ DISEASE "LEUK"))
        OVER PATID SEX)
```

This supplies us with the needed rule. As a bonus, we can avoid having to recompute it later by simply attaching it to the role in the normal way. The similar computation of rules for complex concepts and roles which are already in the domain comes for free.

3.5 Conclusions, Implementation Status and Further Work

As of this writing, we have incorporated NIKL into the implementation of our natural language question-answering system, IRUS. NIKL is used to represent the knowledge in a Navy battle-management domain. The simplification transformation described in this paper has been implemented in this combined system, and the axiomatization of the database as described above is being added to the domain model. At that point, the methodology will be tested as a solution to the difficulties now being experienced by those trying to write the translation rules for the complex database and domain of the Fleet Command Center Battle Management Program of DARPA's Strategic Computing Program.

I have presented a limited inferenc method on predicate calculus expressions, whose intent is to place tnem in a canonical form that makes other inferences easier to make. Metaphorically, it can be regarded as "sinking" the expression lower in a certain logical space. The goal is to push it down to the "level" of the database predicates, or below. We cannot guarantee that we will always place the expression as low as it could possibly go - that problen is undecidable. But we can go a good distance, and this by itself is very useful for restoring the tractability of the mapping transformation and other sorts of deductive operations [10].

Somewhat similar simplifications are performed in the work on ARGON [6], but for a different purpose. There the database is assumed to be a full, rather than a partially specified, model and simplifications are performed only to gain an increase in efficiency. The distinguishing feature of the present work is its operation on an expression in a logical language for English meaning representation, rather than for restricted queries. A database, given the purposes for which it is designed, cannot constitute a full model for such a language. Thus, the terminological simplification is needed to reduce the logical expression, when possible, to an expression in a "sub-language" of the first for which the database *is* a full model.

An important outcome of this work is the perspective it gives on knowledge representation systems like NIKL. It shows how workers in other fields, while maintaining other logical systems as their primary mode of representation, can use these systems in practical ways. Certainly NIKL and NIKL-like systems could never be used as full meaning representations - they don't have enough expressive power, and were never meant to. This does not mean we have to disregard them, however. The right perspective is to view them as attached inference engines to perform limited tasks having to do with their specialty - the relationships between the various properties and relations that make up a subject domain in the real world.

Acknowledgements

First and foremost, I must thank Remko Scha, both for valuable and stimulating technical discussions as well as for patient editorial criticism. This paper has also benefited from the comments of Ralph Weischedel and Jos De Bruin. Beth Groundwater of SAIC was patient enough to use the software this work produced. I would like to thank them, and thank as well the other members of the IRUS project - Damaris Ayuso, Lance Ramshaw and Varda Shaked - for the many pleasant and productive interactions I have had with them.

References

[1] Bates, M. and Bobrow, R.J. A Transportable Natural Language Interface for Information Retrieval. In *Proceedings of the 6th Annual International ACM SIGIR Conference*. ACM Special Interest Group on Information Retrieval and American Society for Information Science, Washington, D.C., June, 1983.

[2] Brachman, R.J., Fikes, R.E., and Levesque, H.J. Krypton: A Functional Approach to Knowledge Representation. *IEEE Computer, Special Issue on Knowledge Representation* , October, 1983.

[3] Codd.E.F. A Relational Model of Data for Large Shared Data Banks. *CACM* 13(6), June, 1970.

[4] Grosz, B.J., Appelt, D.E., Martin, P. and Pereira, F. *TEAM: An Experiment in the Design of Transportable Natural-Language Interfaces*. Technical Report 356, SRI International, Menlo Park, CA, August, 1985.

[5] Moser, M.G. An Overview of NIKL, the New Implementation of KL-ONE. In Sidner, C. L., et al. (editors), *Research in Knowledge Representation for Natural Language Understanding - Annual Report, 1 September 1982 - 31 August 1983*, pages 7-26. BBN Laboratories Report No. 5421, 1983.

[6] Patel-Schneider, P.F., H.J. Levesque, and R.J. Brachman. ARGON: Knowledge Representation meets Information Retrieveal. In *Proceedings of The First Conference on Artificial Intelligence Applications*. IEEE Computer Society, Denver, Colorado, December, 1984.

[7] Scha, R.J.H. English Words and Data Bases: How to Bridge the Gap. In *Proceedings of the 20th Annual Meeting of the Association for Computational Linguistics*, pages 57-59. Association for Computational Linguistics, June, 1982.

[8] W.J.H.J. Bronnenberg, H.C. Bunt, S.P.J. Landsbergen, R.J.H. Scha, W.J. Schoenmakers and E.P.C. van Utteren. The Question Answering System PHLIQA1. In L. Bolc (editor), *Natural Language Question Answering Systems*. Macmillan, 1980.

[9] Stallard, D. Data Modelling for Natural Language Access. In *The First Conference on Artificial Intelligence Applications*, pages 19-24. IEEE Computer Society, December, 1984.

[10] Stallard, D.G. *Taxonomic Inference on Predicate Calculus Expressions*. Technical Report, BBN Laboratories Inc., 1986. In preparation.

END

DATE

FILMD

3-88

DTIC